Event Detection in Twitter With an Event Knowledge Base

Luis Rei, Marko Grobelnik and Dunja Mladenić
Jožef Stefan Institute and Jožef Stefan Postgraduate School
Jamova cesta 39, 1000 Ljubljana, Slovenia
Tel: + 386 14773528; fax: + 386 14773851
e-mail: {luis.rei, marko.grobelnik, dunja.mladenic}@ijs.si

ABSTRACT

We present two methods for event detection in *Twitter* using an event knowledge base. The knowledge base used contains world events reported in the media that we identify as multi-lingual clusters of mainstream news stories. Given this fact, we reduce the problem of event detection to matching tweets to mainstream news stories. The first method consists of using URLs to mainstream news sites present in tweets and in the knowledge base. We use this method to build a supervised corpus of tweets and then create and evaluate a supervised classifier as our second method. Experimental evaluation on real-world data shows that the proposed methods perfom well on our dataset.

1 INTRODUCTION

Twitter is a microblogging social network service with 316 million monthly active users who together generate an average of 500 million messages called tweets per day as of June 2015[1]. *Twitter* users publish tweets about any topic and in any language they choose with a limit of 140 characters per tweet. Because of its large number of active users, it's massive volume of data and the fact that most published tweets are publicly accessible as opposed to other social networks where messages are traditionally restricted to friends, *Twitter* is often used for research.

The definition of event as been subject to academic discussion with different authors adopting slightly different definitions. It is generally agreed upon that an event should be defined as a real-word occurrence over a specific period of time and in a specific location [1]. We adopt that definition and restrict this work to *significant events* as defined in [1] where an event is significant if it may be discussed in traditional media.

The problem of event detection in streams has often been tackled using stream clustering and topic modelling techniques [2]. Stream clustering is the approach used by Event Registry[2] [3]. Social Media streams in general and *Twitter* in particular pose a bigger challenge to traditional event detection techniques: 1) high volume 2) a high degree of non-relevant messages *("meaningless bables")* [4] 3) reduced context for textual based methods as social media messages are usually much shorter than tradional news articles, with *Twitter* messages being limited to 140 characters. Our approach instead relies on the existence of an event knowledge base. Event Registry is one such knowledge base, automatically created from news articles retrieved by newsfeed [5] which collects content from more than 100,000 news sources worldwide with between 100,000 and 150,000 news articles collected daily. Events in Event Registry consist of a multi-lingual cluster [6] of news articles as well as information extracted from them such as named entities, categories and keywords. Since most of the topics discussed on Twitter are also mainstream news [7] and this also corresponds to our definition of significant, the choice of knowledge base seems optimal. Furthermore, the multi-lingual nature of the event information helps us to create mostly language independent multi-lingual methods. Finally, once we match a tweet to an event we can immediately obtain more context for the event. The obvious downside is that we can only detect events already present in the knowledge base.

In Section 2 we present our URL based matching strategy while in Section 3 we present our content based supervised classifier method. Section 4 contains the details of our dataset. Section 5 contains the results of our supervised classifier and Section 6 our final remarks.

2 URL MATCHING

In URL based matching we look for URLs in tweets and compare them to URLs in our knowledge base from Event Registry. If a tweet contains a URL that matches the URL of an article in an event, we can say that the tweet is related to that article and thus to the event that contains the article.

This task is made slightly more challenging than simple string matching by the fact that the relationship between an article and a URL is often one-to-many. The most visible case is when URL shorteners are employed, where a different shorter domain is used in conjunction with a short code which then redirects to the longer URL[3]. Another case is when the URL for the article changes and the old URL redirects to a new one using the HTTP response status code 301 Moved Permanently [8]. It is also common for URLs to contain tracking query strings such as

[1] Twitter, https://about.twitter.com/company
[2] Event Registry: http://eventregistry.org

[3] For example, *http://alj.am/1OAQ9K9* directs the user who clicks on that link to *http://america.aljazeera.com/articles/2015/3/29/nashvilles-boom-pricing-out-middle-and-lower-class.html*.

http://www.example.org/1?utm_campaign=ex or query strings that specify viewing options such as *http://www.example.org/1?page=1*. Furthermore publishing software often makes the same content available under different URLs based on the category structure, e.g. *http://example.com/politics/new-economic-policy/* and *http://example.com/economy/new-economic-policy/* . As a final example, several string level differences can be configured to be ignored by web server software to allow users to reach the right content even when they do not enter the exact string into their browsers, e.g. http://www.example.com/article can be the point to the same article as http://www.example.com/article/ .

To avoid being penalized in search engines rankings for content duplication, many publishers implement either HTTP redirection or the canonical link element [9]. The canonical link element commonly referred to as the canonical tag, is an HTML *<link>* element with the attribute *rel="canonical"* that can be inserted into the <head> section of an article (or any web page) e.g. <link rel="canonical" href="http://example.org/article/new-economic-policy/" />. It is also possible for the canonical link to be present in the HTTP headers instead of the HTML source. It is also common for publishers to implement the Open Graph Protocol [10] which allows for better integration with Facebook and requires a *<meta>* tag with the property *og:url* containing the article's canonical URL.

For each URL in newsfeed and in a tweet we make a request to it, taking note of any redirection, analyzing the headers and processing the HTML response body to attempt to obtain some of the most likely alternative URLs for the content. Each article is associated with a list of URLs used to reference it and each tweet is associated with a list of URLs mentioned in it. We use these lists to match the two. The order of precedence used is:

1. canonical tag/header;
2. open graph *og:url* property;
3. redirection;
4. the original URL;
5. the original URL with or without a trainling slash depending on wether it was not originally present or if it was.

We have opted at present not to address other issues with URLs such as removing query parameters and other URL normalization techniques that can introduce false positives.

3 CONTENT MATCHING

Not all tweets that refer to an event will include a URL to a news story about the event. Thus a different strategy is necessary to match those tweets to events. This is accomplished based on textual similarity between the tweet and events.

Each event within Event Registry contains a cluster of news articles, we select the medoid news article for each language in the event, i.e. the most representative news article for a given language, and use its text to compare to the tweet in that language. If the event does not have a news article in the tweet's language, it is not considered for matching with that tweet.

The problem of matching an article and a tweet is treated as a supervised binary classification problem where given an article and a tweet our classifier must answer if they 'match' or not.

3.1 Preprocessing and Feature Extraction

Text in articles and tweets is preprocessed similarly. Each document is preprocessed according to the following steps:

1. converted to lower case;
2. all URLs are removed;
3. all non-alphanumeric characters are removed (including punctuation and the hashtag symbol);
4. all characters are converted to their Unicode normal form [11];
5. the text is tokenized based on whitespaces;
6. stopwords are removed.

All tweets which after this preprocessing have less than 4 tokens are discarded. Once a document has been preprocessed, we generate its unigrams, bigrams trigrams and quadgrams i.e. its n-grams where $n \in [1, 4]$. For news articles, the title and the body are processed separately.

For each article-tweet pair and for each $n \in [1, 4]$ we generate a similarity vector containing different measures of similarity between their n-grams [12]:

1. the Jaccard similarity between the title of the article and the tweet;
2. the number of common terms between the tweet and the body of the article multiplied by logarithm of the number of terms in tweet;
3. the Jaccard similarity between the body of the article and the tweet;
4. the cosine similarity between the body of the article and the tweet.

3.2 Classifier

We used a linear Support Vector Machine (SVM) as our binary classifier and then performed a random 50-50 split on the dataset into development and test subsets. Using precision as our scoring function, we performed parameter tuning using grid search with 5 fold cross evaluation on the development set for the penalty parameter (C) and class weight hyper-parameters, arriving at C=10 and a positive class weight 0.6 (negative class weight was kept fixed at 1). The positive class weight value multiplies C, since it is lower than 1 it allows the SVM to learn a decision function that makes more misclassifications of positive examples. In particular, more false negatives.

3.3 Language Dependencies

While this classifer is mostly language independent, a few aspects are note. Chief among them is the whitespace based tokenization which while we can expect to work equally well across European languages, will not work at all for Asian languages that do not use whitespaces.

The next, more subtle problem is Unicode normalization. We can expect it to perform better in language in which this normalization corresponds to the way people write on social media than in languages where it does not. For example, in languages which use graphical accents, this normalization step removes them and uses simply the corresponding vowel or consonant. It is common for social media users to also eschew the use of graphical accents. In French and Portuguese, for example, this normalization step matches social media users exactly. However in German, Umlaute are instead commonly replaced in social media with the corresponding vowel followed by an "e". So ä is replaced by ae, ö by oe. This does not match Unicode normalization and thus we can expect worse results from this performing this step in German than we would in Portuguese.

The final language dependency is stopword removal. We rely on the availability of stopword lists compiled by language experts. These may not be available for all languages and/or their quality can vary. It is theoretically possible to generate these lists automatically however we have not taken this step or performed any comparison.

4 DATASET

In order to treat our problem as a supervised classification problem we must first create a supervised dataset. The URL matching described in Section 2 was used on historical data to create the positive examples dataset. The negative examples are generated by pairing tweets that have been matched by this method to a specific event with a different event. We discarded from the dataset any article-tweet pair with a zero similarity vector in both positive and negative examples except for 1 in the negative examples. The number negative examples generated matches the number of positive examples used *i.e.* the dataset is balanced. The total number of examples in the dataset we generated was 32,372 tweet-event pairs.

This dataset generation process supports our goal of obtaining a high precision classifier: the classifier is trained almost exclusively with the hard cases: negative examples that share some similarity with the article. In practice these are actually an extremely small minority of all possible negative examples, since most tweets do not share any similarity with a given article. It also underlies the fact that by relying on simple textual similarity between a tweet and a single article in an event ensures that many true positives are disregarded since they will also have a 0 similarity vector.

4 RESULTS

Our classifier obtained an AUC score of 0.91 on our dataset. The Precision-Recall curve is shown in Figure 1*: Precision Recall Curve* and the Precision-Recall vs Threshold curve is shown in Figure 2*: Precision-Recall vs Threshold*.

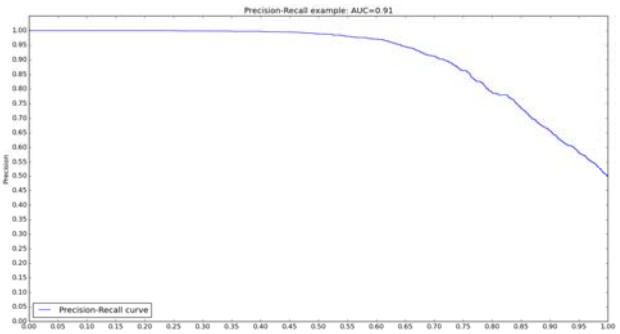

Figure 1: Precision Recall Curve

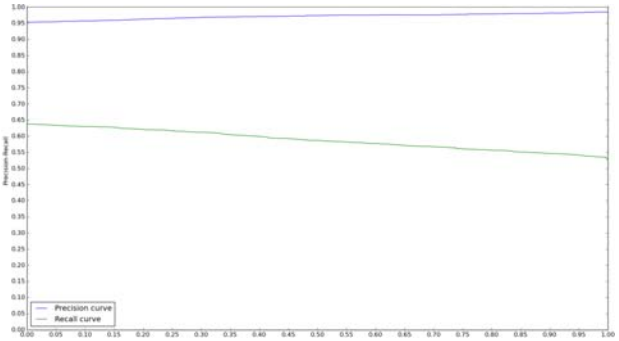

Figure 2: Precision-Recall vs Threshold

We can see that if we chose a threshold near 1, our classifier has nearly 100% precision while lowering our classifiers recall to nearly 55%. The procedure for generating our dataset introduces a huge bias into our evaluation: in reality, we will have many more false negatives since many true positives have zero similarity vectors and will thus become false negatives (those cases were discarded in our dataset). Real recall can be expected to be much lower than the recall on our test dataset. We can however expect that this will be partially offset by an expected redundancy in social media messages and a large volume of messages regarding events. The emphasis on precision over recall in our work is also understandable in the context of future applications. The end use of such a classifier is likely to be either to directly show tweets to end users of a web site or to give a social dimension to the analysis of events. In either case, the loss of tweets from a sample seems preferable to either showing the wrong tweets to an end user or to reduce the accuracy of social media analysis with respect to events under analysis.

6 FINAL REMARKS

Event Registry adds between 5000 and 40000 events to its database every day. Considering also the daily volume of tweets, even if we consider only the public twitter stream which contains only 1% of all tweets, we are looking at an estimated lower bound of 25M daily possible tweet-event pairs. This number becomes considerably more problematic if we add a reasonable window of 6 days around the

publishing of a tweet when considering which events to match it to. While URL matching is computationally cheap and a classification algorithm can also be considered computationally cheap, feature extraction is not so cheap. Thus, running a classifier against event-tweet pairs in practice should be restricted to a subset of all possible event-tweet pairs that are considered good candidates. Fortunately, since the classifier relies exclusively on textual similarity, we can rely on decades of research and development in Information Retrievel and databases to provide a set of good candidates efficiently.

We consider the biggest contributions of this work to be the use of a knowledge base for event detection in social media and the introduction of a fully automated technique for generating a supervised event detection dataset.

ACKNOWLEDGMENTS

This work was supported by the Slovenian Research Agency and the ICT Programme of the EC under Symphony (FP7-ICT- 611875).

References

[1] A. J. McMinn, Y. Moshfeghi and J. M. Jose, "Building a large-scale corpus for evaluating event detection on twitter," in *Proceedings of the 22nd ACM international conference on Conference on information \& knowledge management*, 2013.

[2] C. C. Aggarwal and K. Subbian, "Event Detection in Social Streams," *SDM,* vol. 12, pp. 624--635, 2012.

[3] G. Leban, B. Fortuna, J. Brank and M. Grobelnik, "Event registry: Learning about world events from news," in *Proceedings of the companion publication of the 23rd international conference on World wide web companion*, 2014.

[4] J. Weng and . B.-S. Lee, "Event Detection in Twitter," in *ICWSM*, 2011.

[5] M. N. B. Trampuš, "Internals Of An Aggregated Web News Feed," in *SiKDD*, Ljubljana, Slovenia, 2012.

[6] J. Rupnik, A. Muhic and P. Skraba, "Multilingual Document Retrieval Through Hub Languages," in *Proceedings of the Fifteenth International Multiconference Information Society*, 2012.

[7] H. Kwak, C. Lee, H. Park and S. Moon, "What is Twitter, a social network or a news media?," in *Proceedings of the 19th international conference on World wide web*, 2010.

[8] R. G. J. M. J. F. H. M. L. B.-L. T. Fielding, "Request for Comments: 2616: Hypertext Transfer Protocol -- HTTP/1.1," Network Working Group, The Internet Society, 1999. [Online]. Available: https://tools.ietf.org/html/rfc2616. [Accessed 2015 March 10].

[9] M. K. J. Ohye, "Request for Comments: 6596," Internet Engineering Task Force (IETF), April 2012. [Online]. Available: http://tools.ietf.org/html/rfc6596. [Accessed 15 March 2015].

[10] Facebook, "The Open Graph Protocol," 20 October 2014. [Online]. Available: http://ogp.me/. [Accessed 10 March 2015].

[11] M. Davis and K. Whistler, "Unicode Standard Annex 15: Unicode Normalization Forms," The Unicode Consortium, 2015.

[12] P. Saleiro, L. Rei, A. Pasquali, C. Soares, J. Teixeira, F. Pinto, M. Nozari, C. Felix and P. Strecht, "POPSTAR at RepLab 2013: Name ambiguity resolution on Twitter," in *CLEF 2013 Eval. Labs and Workshop Online Working Notes*, 2013.